Association for Information Systems
AIS Electronic Library (AISeL)

AMCIS 1996 Proceedings

Americas Conference on Information Systems (AMCIS)

8-16-1996

A Genetic Algorithmic Approach for Solving the Multi-Resource General Assignment Problem: Design and Configuration Issues

Lawrence J. Schmitt
Department of Information Technology Management, Christian Brothers University, lschmitt@odin.cbu.edu

A Genetic Algorithmic Approach for Solving the Multi-Resource General Assignment Problem: Design and Configuration Issues

Lawrence J. Schmitt, Department of Information Technology Management,
Christian Brothers University; 650 East Parkway South, Memphis, TN 38104; (901)722-0571;
Internet: lschmitt@odin.cbu.edu

Introduction

The potential of Genetic Algorithmic (GA) approaches to solve complex problems has been recognized in recent research. Building on the configuration and design recommendations developed in a previous comprehensive evaluation of GA to solve the TSP (Schmitt 1994, Schmitt and Amini 1995a), and other studies that developed a GA to solve the VRPTC (Schmitt 1994, Schmitt and Amini 1995b), an effective Genetic Algorithmic heuristic to solve Multi-Resource Generalized Assignment Problem (MR-GAP) is designed and developed in this study. The primary focus of this work is the design of a Genetic Algorithm to solve the MR-GAP (GA-MRG).

Problem Definition

The Multi-Resource Generalized Assignment Problem (MR-GAP) is an extension of the Generalized Assignment Problem (GAP). In the GAP (Fisher & Jaikumar 1981) an agent is assigned to multiple tasks subject to the availability of a single resource that is consumed by the agent in accomplishing the task. The MR-GAP extends this concept by assigning each agent multiple resources that are consumed in the accomplishment of the assigned task. This extension is significant in that it allows the modeling of more complex models reflecting the more typical problems faced in real life problems.

The MR-GAP has been used in a wide variety of applications ranging from solving truck routing problems to planning the location of distributed databases. Potential applications include transportation and telecommunications network design, cargo loading, warehouse design, work planning, course scheduling and classroom assignment.

A mathematical formulation of the multi-resource generalized problem is :

MR-GAP

(1)

Subject to:

(2)

, (3)

(4)

Where ***I*** is the index set of tasks, ***J*** is the index set of agents and *K* is the index set of resources used by agents to perform these tasks. a_{ijk} is the amount of resource *k* used by agent ***j*** in performing task **I,** $\mathbf{b_{jk}}$ is the amount of resource ***k*** available to agent ***j*** and $\mathbf{c_{ij}}$ is the cost of assigning task ***I*** to agent ***j***. The decision variables $\mathbf{x_{ij}}$ are defined as 1 if task ***i*** is assigned to agent ***j***; 0 otherwise. A discussion this formulation and of various alternative algorithms to solve the MR-GAP can be found in Gavish and Pirkul (1991**).**

Genetic Algorithm Overview

Genetic Algorithms (GA), the focus of this study, are a relatively new method. Developed in the 1970's by John Holland and his students at the University of Michigan they were originally applied in unconstrained optimization ((Holland (1975) and De Jong (1980)). Approximately ten years later, the original GA was applied to constrained optimization problems (Davis (1991), Goldberg (1989), and Whitley et al. (1989 and 1991)).

The GA is a meta-heuristic that directs a global search of the solution space to find the best possible solution. GAs are a weak method, meaning that they make few assumptions about the problem domain. Weak methods are often low power in that they are not very efficient. Genetic Algorithms represent an apparent anomaly, a powerful weak method. A brief outline of the general GA heuristic is presented below:

CREATE* and *EVALUATE an initial population**.**

/* Evolve population until complete */

WHILE stopping criteria not met

Select ***Parents*** using selection method.

Perform ***Crossover*** with probability *x_rate.*

Perform ***Mutation*** with probability *mu_rate*.

Perform ***Evaluation.***

IF *ga_type* is generational, insert new gene into off-line population using replacement strategy.

OTHERWISE; insert new gene directly into population using replacement strategy.

IF *ga_type* is generational and off-line population is same size as on-line population swap populations.

IF elitist replacement strategy ensure that best of old population is moved to new population.

END WHILE

Collect ***Summary Data*** and output files when each replication is completed.

END

GA Design Considerations

The GA is an attempt to solve complex problems using simple building blocks. Modeled after general evolutionary theory, this approach seeks to emulate the robustness found in natures survivors.

GAs concurrently develop a population of potential solutions to the given problem. GAs are somewhat unique in that to solve a given problem, it must be able to evaluate potential solution but, it does not necessarily require knowledge of how to solve the problem.

The first design issue is to select an encoding that will capture the critical characteristics of the problem to be solved. In GA terms, this is developing the genotype.. The chromosome is the string of bits or integers that encode the solution. Each bit or string is a gene and the value of a given gene is commonly called an allele. To solve the MR-GAP, the critical problem characteristic is the assignment of agents to tasks. The encoding chosen for the chromosome is a vector of integers in which each vector location represents a task that must be performed. The gene contains the numerical code identifying the agent assigned to accomplish the task assigned to this vector location. Additional information required to solve the problem such as the quantity of each resource required for each task and the amount of each resource type currently available to a given agent are kept in supporting data structures.

An initial population of chromosomes is developed by randomly assigning a number from 1 to the maximum number of agents to a given vector location or task. During the generation process, each chromosome is evaluated to assess it's fitness.

The process of evolving a solution begins by selecting promising chromosomes based on their fitness using a pre-defined selection method. The chromosomes are mated by applying a crossover function to them. They are then exposed to the probability of mutation. This process allows the combination of building blocks or schemata to build potentially better chromosomes.

Two general method are utilized for evolutionary population control. The steady state approach (Whitley 1988) inserts newly created chromosomes directly into the population replacing other less fit chromosomes. The alternative approach generational, creates an entire new population which replaces the original population when the required number of chromosomes are generated. Earlier studies (Schmitt 1994, Schmitt and Amini 1995a, Schmitt and Amini 1995b), achieved superior results using this method of evolutionary control. Consequently steady state evolutionary control is utilized in this study.

To determine the fitness of each chromosome, an evaluation function is developed. This fitness evaluation function calculates the total cost of assigning each task to the selected agents. A penalty for infeasible solutions is calculated by determining the number of infeasible assignments of agents to tasks and multiplying the cost of the solution by this factor..

Generating new chromosomes requires the selection of high potential candidates to be parents, and creating the new chromosome by mixing their genetic material using a crossover process. Several methods of selecting parent chromosomes exist. Based on prior experience, this study uses a ranked based selection scheme based on the fitness of individual chromosomes.

The study of crossover operators has been a particularly prolific area of GA research. This study utilizes a simple crossover function in which a location on the chromosome is randomly selected and new chromosomes are created by taking the gene values from the beginning of the gene to this location from parent 1 and the balance from the location to the end of the gene from the parent 2. Two new chromosomes are created during this process by allowing each of the selected chromosomes to be the first or second parent.

Random error is introduced to mimic the effects of nature on existing lifeforms. A randomly applied swap mutation function is created to mimic this effect.. Basically the swap mutation function selects two genes on a chromosome and swaps their values.

The final step in the evolutionary process is to insert the new chromosome in the population. As the population is a fixed size the least fit chromosome in the population is selected to be replaced. This replacement strategy is inherently elitist and often has the effect of forcing premature convergence. Preliminary testing does not seem to show premature convergence in this case.

The final design issue is to establish a stopping criteria. Initially convergence is utilized. Convergence is when all member of the population give the same solution to the problem. Some obvious problems with this approach is that it tends to require much more computing resources than necessary It appears that this may not be the best stopping criteria.

GA - MRGAP CONFIGURATION

Population Creation	Randomly Generate
Population Size	#Tasks * # Agents
Evolutionary Strategy	Steady State
Selection	1.6 Biased Rank
Replacement	By Rank
Mutation	15% Swap
Crossover	60% Crossover

Initial Testing

To evaluate the viability of GA-MRG in solving MRG problems 36 sets of 10 problems each were randomly generated. These problems varied in class, size from 3 to 10 agents and from 50 to 100 tasks and density from 70% to 100 %. Prior to each execution a lower bound solution for each problem was calculated by relaxing the resource constraint for each agent. This lower bound was utilized to compare the quality of solution that GA-MRG obtained. On all problems the results showed that GA-MRG found solutions from 0 to 20% from the theoretical lower bound. In many cases the lower bound was infeasible. Where the lower bound was infeasible the gap between GA-MRG and the lower bound tended to be greater. In most cases GA-MRG found a feasible solution to the problem. CPU time varied from .01 seconds to 14 seconds on a P5-60 running Linux 1.3.70.

Summary

This paper presents a viable GA based solution to the MR-GAP problem. Additional emperical testing and tuning is required on this implementation. Preliminary results show that GA-MRG is competitive with other more traditional approaches in terms of solution quality and CPU utilization. It appears that the final solution appears generally within 20% of the generations required for convergence.. Additional efforts to leverage this may lead to further improvements in performance.

Further research in this area will include tuning and hybridization of the GA with other techniques such as Tabu Search as well as testing of this technique on solving practical business applications.

Bibliography available upon request from author.

BIBLIOGRAPHY

Davis, L. (1991). Handbook of Genetic Algorithms. New York: Van Nostrand Reinhold.

De Jong, K. (1980). Adaptive system design: A genetic approach. IEEE Transactions on Systems, Man and Cybernetics, 10(9), 556574.

Fisher, M. & Jaikumar, R. (1981). A generalized assignment heuristic for vehicle routing. Networks, 11, 109124.

Gavish, B. and H. Pirkul (1991). Algorithms for the Multi-Resource Generalized Assignment Problem, *Management Science vol 37 no 6,* (pp 695-713).

Goldberg, D.E. (1989). *Genetic Algorithms in Search, Optimization and Machine Learning*, Addison-Wesley Publishing Company, Inc., Reading.

Holland, J. (1975). Adaptation in Natural and Artificial Systems. MIT Press, Cambridge, MA.

Schmitt II, L. J. (1994). *An Empirical Computational Study of Genetic Algorithms to Solve Order Based Problems: An Emphasis on TSP and VRPTC*, Ph.D. thesis, MIS/DS Department, The University of Memphis.

Schmitt, L. J., and M.M. Amini (1995a). *Genetic Algorithmic Approaches to the Traveling Salesman Problem: Design and Configuration Issues*, Working Paper ITMGA01.002/95, Christian Brothers University, Memphis, TN.

Schmitt, L. J., and M.M. Amini (1995b). *Performance Characteristics of Alternative Genetic Algorithmic Approaches to the Traveling Salesman Problem: A Rigorous Empirical Study* Working Paper ITMGA02.008/95, Christian Brothers University, Memphis, TN.

Whitley, D. (1988). Genitor: A different genetic algorithm. Technical Report CS88101, Colorado State University.